DISTRIBUTED AGENT SEARCH EFFICIENCY

An Undergraduate Research Scholars Thesis

by

CHRISTOPHER MURRAY

Submitted to Honors and Undergraduate Research
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Research Advisor: Dr. Thomas Ioerger

May 2015

Major: Computer Science

TABLE OF CONTENTS

ABSTRACT

Distributed Agent Search Efficiency. (May 2015)

Christopher Murray
Department of Computer Science
Texas A&M University

Research Advisor: Dr. Thomas Ioerger
Department of Computer Science

Multi-agent systems where there exists a group of agents all working towards the same goal are gaining increased focus. One subset of these systems is groups of agents who have intermittent communication with one another. In an environment with limited communication the group of agents must act individually, while still completing their shared goal. Since each individual agent only knows about its experiences and those of other agents it meets, it is important that a system be developed to facilitate agent collaboration.

This thesis looks at one such application of a multi-agent system with limited agent communication where the agents are tasked with retrieving a set of objects – Distributed Agent Search. Within this application it is important that the agents act as a team even while not in constant contact with each other and that the search strategy used to retrieve the objects is the most efficient for the given environment. Through experimentation it was determined which of a set of developed search strategies performs most efficiently given different environment sizes, and varying communication.

CHAPTER I

INTRODUCTION

It is becoming increasingly important in multi-agent environments that the constituent agents act in a coordinated and directed fashion in order to efficiently complete the task they have set out to accomplish. This is especially true in the field of search, since the group of distributed agents can more effectively seek out and retrieve all objects when working as a team instead of unilaterally. In order for the agents to efficiently retrieve all objects it is necessary that they work independently while still maintaining a belief state of the environment that includes the other agents and what they have accomplished. This allows the individual agents to seek out objects, while still acting in a coordinated manner according to the team's overall goal. Without this sense of coordination and mutual belief about the state of the world, the agents would act unilaterally, with no regard for the team's goal of retrieving all objects and returning home and as such one or more team members could be left stranded, not knowing the goal has been completed.

The simplest representation of a distributed agent search a group of agents are searching for a group of stationary objects, and the agents are in constant communication with each other. In this scenario is it trivial for the agents to maintain a single representation of the world and constantly have the most recent knowledge about all other agent's activity, similar to two baseball players running for the same ball where one player can communicate with the other to inform him that he is going for the catch. This constant communication allows trivial coordination where the agents do not have to have their own belief states and can always act on correct data. However,

in reality the agents may not be able to constantly communicate, such as robots searching a collapsed building, where debris may block an agent's communication. This lack of constant communication makes the agents need to reason about the state of the world, and update their beliefs accordingly when they do come into communication range with another agent. In the example of robots searching a collapsed building it is critical that the robots continue their task, even when not in constant communication with each other, but it is also necessary that they do maintain some form of intermittent communication in order to determine if the team's goal has been accomplished.

In this research project we tackle one implementation of the distributed search problem. The main facets of this problem include a group of agents and a group of objects that the agents must collect. The agents will all start at a single location and then disperse to search for, and collect a set of objects, before returning to the starting location. There are several limitations on the agent's ability to collect these objects, such as limited communication range, constantly moving objects, an agent's ability to carry only one object, the stipulation that an agent must know all objects have been found before returning to the start, and the restriction to pairwise meetings between agents. These limitations are meant to simulate the group of agents searching a destroyed building for survivors, where the survivors may also be moving, and because of the terrain there is limited, intermittent communication between the agents. The main focus of this problem is finding an efficient search strategy, or movement pattern, that results in the agents accomplishing their goal in the shortest amount of time. Achieving this goal will rely on a reliable implementation of the agent's belief state, an efficient way to share this belief state

between agents when they meet, and a belief reasoning scheme that will allow an agent to make the correct decision about when the goal has been successfully accomplished.

The essential features of the distributed agent search problem that our system is replicating and that present the unique challenge in this research are limitations on the agent's sensor range, the non-stationary nature of the objects, and the team-oriented structure of the agents. The limited sensor range imposes several key constraints on the agents which are that agents must be near each other to communicate, as well as near an object to pick it up. This also limits the agent's understanding of the world, since it can only have certainty in its beliefs about what is directly within its sensor range, while the space outside its sensor range can change without the agent's knowledge. Moving objects also pose a unique challenge, since the agents cannot discount any location in the search area as not containing an object and must continue searching the entire search space until all objects have been collected. The primary feature of this problem, however, is the team-oriented structure that the agents exist in. Within this team communication is essential because each agent is limited to collecting only one object – although objects are not pre-assigned to a specific agent – and an agent can only return to the start, or home, when it knows all objects have been collected. However, it is not enough that one agent returns home to complete the task, all agents must return for the task to be accomplished successfully. This requires agents to hold beliefs about other agents and what their current beliefs of the world are. Because of this, an agent cannot simply return home when it knows all objects have been collected, but must be able to reason that all agents hold the belief that all objects have been collected, so that the agent can be sure all other agents will also return home, thus completing the task.

One system of representation for beliefs is truth-maintenance, or TMS. The Truth Maintenance System for representing belief records knowledge about deductions and represents a belief as node structure, which contains the belief as well as the justification for that belief [Doyle, 1978]. When a new justification is presented that alters the set of beliefs the process of truth maintenance is started and all nodes that had a belief that is altered by the new justification will be reexamined [Doyle, 1978]. The TMS representation can also reason about the belief nodes and determine if a node represent a contradiction and also supports queries about consequences or antecedents about a belief, or information that has been derived from beliefs [Doyle, 1978].

Another representation for belief is modal logic, which is a formal logic that provides a way in which to express modality. A system based on modal logic has a knowledgebase of sentences that represent beliefs with both classical propositional logical operators as well as modal operators. These modal operators represent the concepts of necessarily, $\Box$, and possibly, $\Diamond$, and can be combined to form a group of beliefs about the state of the world [Shoham and Leyton-Brown, 2009]. In addition to the unitary modal operators, there are also operators that allow the representation about an agent's belief state. The knowledge operator $Ki(\phi)$ denotes that agent i knows ϕ [Shoham and Leyton-Brown, 2009]. Alongside the knowledge operator there is the belief operator, which is represented as $Beli(\phi, t)$, which means that agent i believes some event ϕ occurred at time t. This distinction between knowledge and belief is necessary because, while related, just because an agent believes an event occurred does not make it necessarily true.

Having a reliable and efficient belief system in place is key as it plays a central role in the agent's ability to perform as a team and complete their task. Each agent in the team has the same task, and must rely on all other agents in the team to complete it; this scenario is described the agents holding a joint persistent goal, or JPG. A JPG is defined by Tambe as a team action p, denoted JPG(θ, p) which requires all team members to mutually believe that p is false and want p to be eventually true [Tambe, 1997]. It is important to represent the agents' goal as a JPG, rather than the conjunction of each agent's goal, as it adds constraints that each individual agent must adhere to. These constraints are that an agent cannot believe a task is complete merely because it believes all objects have been found, but must also believe all other agents believe this as well; this is the mutual belief that all objects have been found. Formally a JPG (JPG x y p q) is defined as (MB x y ¬p) ∧ (MG x y p) ∧ (UNTIL [(MB x y p) ∨ (MB x y □¬p) ∨ (MB x y ¬q)] (MG x y p)), where (MG x y p) is defined as (MB x y (GOAL x ◇p) ∧ (GOAL y ◇p)) [Levesque et al., 1990]. In this definition both x and y are agents, p is a goal, and q is the justification for needing to complete that goal. Informally it follows that a JPG means that both x and y hold a mutual belief that p is currently false, that it is their mutual goal p will be true, and both x and y will hold this goal until they both mutually believe that p is true (goal completed), that it is no longer possible for p to be true (the goal is no longer able to be completed), or the justification for completing p is false.

The original motivation for this research project lies in the most efficient way to represent the belief structure of the agents and how an agent can know when the team's joint persistent goal has been achieved. In order for this joint persistent goal to be achieved a mutual belief must be held between the set of agents, which involves an agent knowing about which other agents also

know the same information as it knows. In order for the goal to be achieved there must be a mutual belief among the agents that all objects have been found. This leads to an infinite conjunction of nested beliefs, since an agent must know that all objects have been found as well as knowing all other agents know all objects have been found and that each agent knows all other agents know all of this information.

A major limitation to using this system of joint persistent goals and mutual beliefs among the agents is the pairwise meetings that the agents participate in. This problem with agent communication is related to the Byzantine General's Problem. The original Byzantine General's Problem is formulated as such: generals of the Byzantine Empire's army must all decide unanimously if they are going to attack an enemy army, however, communication is difficult and the generals must communicate by sending messengers to one another. In addition to sending messengers there are traitors among the generals who can act arbitrarily in order to achieve their own goals. The traitors are successful if the generals do not agree on a unanimous decision, or agree on a decision contrary to the general's desires. Several of the difficulties faced with finding an efficient distributed agent search strategy are reflected in this problem. The distributed nature of the generals makes communication only occur through messengers in pairwise meetings, much like our agents; the lack of confidence in the information presented in a meeting mimics the uncertainty an agent has when determining the belief states of other agents; and the goal of all agreeing unanimously on a course of action, which in the agent's case is the ability to return to the start.

A more generalized version of the Byzantine General's Problem, which better shows the complexity of pairwise interaction, is given as the Two General's Problem. This problem is formulated as such: there are two generals, each with an army that wishes to attack a city. Both must decide on the specific time of the attack or they will surely fail. The two generals are separated by a valley that is filled with hostile forces, and the only way to communicate is to send messengers across to the other. In order for the generals to agree on an attack plan one must propose a plan and the other must send an acknowledgement. Because of the possibility of either the messenger carrying the initial attack plan or the messenger carrying the acknowledgement to the attack plan could be captured before the message is delivered there is no way to guarantee mutual agreement between the generals that the proposed attack plan should be carried out. This more closely resembles the pairwise interaction among the agents and illustrates the impossibility of establishing true mutual belief among more than two agents when limited to pairwise interaction. An example of this is as follows: there are a group of three agents – A, B, and C – and one object, S. Agent A collects object S and has a meeting with agent B; now both agents A and B belief that S has been collected and that the other agent also believes this. Agent B then meets with agent C, and now B and C believe that all agents know S has been collected, however B and C do not believe that A knows all agents believe S has been collected, since A does not believe C believes this. The naïve way to solve this is to conclude that an agent can only return home as long as it believes all other agents believe all agents believe the object has been found. This leads the agents into an infinite loop of meetings caused by one agent not being present in every meeting. This agent that is left out of the pairwise meeting will not possess the same belief state as the other two agents and will therefore need to be contacted so the other

agents can return. This problem repeats indefinitely as only pairwise meetings are possible and all agents cannot simultaneously unify belief states.

Our approach to finding an efficient solution to this problem, where agents can only intermittently communicate in a pairwise fashion, involves creating and testing several search strategies that the agents will use to complete the search task. The first search strategy is a Periodic Synchronization model where all the agents periodically return to a designated meeting point to share the knowledge they have gained since the last meeting. The next strategy is a the Circulating Coordinator model where a subset of the agents spreads evenly around the search space, and the remaining agents circulate between these smaller search spaces propagating knowledge amongst all agents. The third strategy is a Chain model where the search space is evenly partitioned for each agent, and where each partition slightly overlaps its neighboring partitions such that intermittently the agents in neighboring partitions will communicate with each other.

Each of the proposed search models possesses strengths and weaknesses that must be addressed. The Periodic Synchronization model will ensure that knowledge is updated amongst the agents, since they are required to meet at specific intervals during the search process, however this has the drawback of being inefficient, since all agents must return to a singular point before continuing the search. The Circulating Coordinator model seems to be slightly more efficient than the Periodic Synchronization model since it leave a majority of the agents in continual search of the objects, however there is no guarantee the agents moving around attempting to circulate knowledge will ever meet with an agent and this could cause certain agents to possess

an outdated state of the world. The last proposed search strategy, the Chain model, attempts to remedy both of the previous search models by having all agents in search of objects, while partitioning the search space in such a way that communication between agents is likely to happen. The downside to this approach is the overlap between agent search spaces may be required to be so large as to facilitate regular meetings between agents that it becomes inefficient.

CHAPTER II

SEARCH STRATEGIES

The main goal of this research is to determine which of a set of search strategies will complete the agent's given goal in the shortest amount of time, and as such it is important to note the distinctions in possible search strategies, and the search strategies that have been developed for this project.

Search strategies can be broken down into one of two groups: centralized or decentralized. In decentralized search strategies agents act independently of each other and each search in any area of the environment. In centralized search strategies the agents have a predetermined area of the environment in which they are to search, or have a predetermined role that they must fulfill while completing their task. However, even with these two categories not every search strategy can be neatly categorized into just one, and may contain elements of both. The spectrum of this can be seen as the difference between a completely random search strategy, where every agent moves in a random nature, to a perfectly centralized strategy, where one agent directs all other agents and informs each what to do.

The environment being simulated for this research tests both centralized and decentralized strategies and determines if either fairs better in certain conditions than the other. The simulation environment specifically tests how efficient a search strategy is with varying environment size and communication range. This shows which search strategy is most affected by limited

communication, as well as how efficiently the environment is searched by a limited group of agents.

The search strategies being tested in this research fall on the spectrum between centralized and decentralized and can be broken up into two main groups: those that primarily use a random walk for all decision making, and those where agents have a specific role.

At the most basic level a search strategy is simply a movement pattern that the agents use in order to, hopefully, aid them in object collection. The five search strategies are Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

Random Walk

The random walk strategy is the base case used for comparison against the other search strategies. This is a decentralized strategy where movement is determined by a random number generator in which a direction is determined and the agent moves.

It is expected that this strategy will be able to complete the task at smaller environment sizes, but will be heavily affected by decreases in sensor range and increases in environment size.

Periodic Synchronization

In the periodic synchronization model all agents return to a specified location in the world at specific time intervals in order to synchronize their meeting graphs, but at all other times move

using Random Walk. This strategy is slightly more centralized than the pure Random Walk, and it is hoped by doing this that all agents are together, or close to a scheduled meeting when the goal is determined to be accomplished and can complete it in less time than a simple random walk. After the synchronization is complete all agents return to using the Random Walk method for determining movement. This is arguably more efficient than a pure random walk because even if agents are widely spread and randomly moving when one agent deems the goal complete and begins the process of goal completion, all agents will still be periodically meeting at a specified location, so at most the agents will take one full cycle between meetings to all mutually believe the goal is complete.

It is expected that this search strategy will perform better than the pure Random Walk strategy and be able to handle larger environment, but it will still be highly affected by decreases in sensor range, and increasing environment sizes.

Circulating Coordinator

In the circulating coordinator strategy all agents but one is given a roughly even partition of the world to search and the remaining agent is used to move between each of these partitions in an attempt to propagate information and beliefs. This strategy is a more centralized strategy where each agent has its role in the team pre-assigned. In this manner it is hoped that beliefs are propagated more rapidly since one agent is constantly moving between other agents, and will not leave the search environment until all other agents believe the goal is complete. The downside to this method is that the circulating agent may rarely meet other agents depending on each individual agent's search space. Each agent moves randomly around its search space, while the

coordinator moves directly between the centers of each search space in sequence until it reaches the last search space where it will reverse its travel path and trace its route back to the first search space.

It is expected that this strategy will perform significantly better than both of the more decentralized strategies and be able to perform well regardless of environment size, but will most likely still be affected by decreasing sensor ranges.

Chain Model

The chain model search strategy attempts to modify the previous strategy in a way that both facilitates meetings between agents, while allowing full time search. This strategy partitions the world in such a way that all agents receive a search space, but the edges of each search space are overlapping with the neighboring search spaces. The hope is that even though the agents are moving randomly, they are contained within their search space and will occasionally meet other agents in the overlapping regions. The only time an agent may leave their designated search space is when they know that all the goals have been collected. The drawback to this method is that if the overlapping regions are not large enough then meetings will happen infrequently, possibly even less than a pure random walk.

It is expected that this search strategy will perform equally as well as the Circulating Coordinator strategy, but will be more affected by larger environment sizes and decreasing sensor range.

Coordinator-Synchronization

The coordinator-synchronization strategy attempts to merge the Circulating Coordinator and Periodic Synchronization search strategies, so that the agents are directed in their search and are spread around the environment and have frequent communication and updating of their beliefs through the use of the coordinator and the periodic meetings.

It is expected that this search strategy will perform better than both the Circulating Coordinator and the Periodic Synchronization model, and will receive only minimal impact with increasing environment size and decreasing sensor ranges.

CHAPTER III

SIMULATION ENVIRONMENT

In order to demonstrate how a group of agents can solve a task in an environment with limited communication, as well as other restrictions, we created a simulation space. This simulation space sets strict world boundaries and contains all agents as well as all objects and is explained in more detail below. The agents that are in the simulation space make their own decisions about movement and goal recognition based on what they believe the current state of the world to be, while the goal objects simply move randomly.

Several problems had to be resolved while constructing the simulation space, such as agent communication, goal recognition, and ensuring no agent is left behind when all objects have been collected. These problems were solved through the use of a meeting graph which we developed in prior research. These solutions allow for the testing of different search and agent movement strategies. Each search strategy is run through the simulation and results about how long it took to complete the goal is recorded.

Simulation Environment

The simulation environment is constructed of an LxL square grid where each cell can contain either one of the X agents, or one of the Y goal objects. The simulation environment, known as the "world", is simply to be a container for the overall simulation. No information is passed to agents that they could not detect in their sensor range. The simulation environment progresses in steps, each of which representing one uniform unit of time. During each step the following series

of events happen: the agents move, the goal objects move, and the agent's perceptions are updated. Movement is considered to happen all at the same time, therefore the agents do not perceive objects until all actors on the board have moved. During the perception step the world gives each agent an updated description of the world for the cells that are within its sensor range and the agent can take an action, such as picking up an object. During this perception stage is also when adjacent agents will meet and share their beliefs about the current state of the world.

It is important to note that the world is not giving the agents any information that is outside the scope of the agent's sensor. In order to simulate distributed agents in an environment with limited communication the only up-to-date copy of the environment is at the world level, while each agent only has an up-to-date view of their immediate area (within their sensor range). As an agent progressively moves around the world their belief can only be certain in the area directly in their sensor range, as any cells outside that range could have changed during the last iteration.

Agent Object

An agent object represents one agent that is part of the team of agents attempting to fulfill the goal of obtaining all the goal objects and returning to the starting point. Each agent is uniquely identified by a name and resides in one cell in the world. There are several limitations imposed on agents in the simulation which include a limited, but defined, sensor range; the ability to only interact and communicate with objects and other agents within the defined sensor range; and a restriction on only being able to carry one object per agent. Along with these limitations the agents also know the confines of the world they inhabit and can reason to stay within the LxL cell grid.

During each iteration of the simulation an agent will have the chance to make a move. An agent may move in one of five ways: up, down, left, right, and remaining stationary. During this movement step the agent will use the current search strategy to determine its movement, which is described below. After all agents and objects have moved during an iteration the agent has its state of the world updated in the confines of its sensor range. During this step the agent perceives its surroundings and interacts with other agents and objects. For each agent within the current agent's sensor range a meeting takes place where the agents update their meeting graphs and their certain beliefs about the state of the world, such as last known locations of objects and other agents. Once an agent has met with all adjacent agents within its sensor range the agent then will collect up to one object within its sensor range. If the agent has previously collected an object then it will be unable to collect another. If an agent has not previously collected an object and there are multiple objects within its sensor range, then only one object will be collected.

At the end of the perception step an agent determines if all objects have been collected. If it is determined all objects have been detected then the agent begins the procedure to return to the starting location as described in the Goal Detection section. If it is determined that all objects have not been collected, then it simply ends the perception step and the simulation continues on to the next agent.

Goal Object

A goal object, or simply an object, is the representation in the simulation of the thing the agents are searching for; in a real-world setting this could be survivors in a collapsed building. During each iteration of the simulation each object makes a movement. Unlike an agent, an object

simply moves randomly around the world, has not internal representation of the world, does not communicate with other objects or agents, and does not have a perception step. Once an object has been collected by an agent it is removed from the simulation and no longer participates during the remaining simulation iterations.

Meeting Graph

The meeting graph is the mechanism in which agents reason about their beliefs and the beliefs of the other agents. A meeting graph is represented as a directed graph containing the full history of all agent interactions including meetings with other agents, object locations, and collected objects. The nodes of the meeting graph represent a meeting between two agents, and the edges represent the set of events that occurred since an agent's last meeting with another agent.

Each node in the meeting graph represents not only a meeting between two agents, but also the set of facts mutually believed by those two agents at that time. As such, each node contains the two agents that were present at that specific meeting, the list of facts that are mutually believed by both agents, and the time that the meeting took place. The time that the meeting takes place is simply the iteration number that the simulation is currently on. An edge in the meeting graph is directed in the direction of time, i.e. it is pointing away from earlier time steps and towards more recent events. Since each edge represents an agent's observations between meetings it is labeled with an agent as well as a set of events. Each event in the set of events consists of what took place and at what time step. It is important that events are labeled with the time step in which they occur. Consider the tracking of an object: if both agent A and agent B observe that object O is at a specific location (with neither agent collecting the object) this could be helpful in agent C

predicting the area the object is currently likely to be in. If both agent A and agent B report object O being located at opposite ends of the world, this does not help agent C, but if the location observations are labeled with the time in which they were observed, then agent C can reason that it is most helpful to search in the area of the most recent siting, thus helping find the object quicker as long as the siting is not too old.

When two agents meet a new node is created that represents this meeting. The set of mutual beliefs associated with this node is the union of the current mutual beliefs of both agents participating in the meeting. In this way a node is given a canonical name based on the agents that participated in the meeting where each agent name is subscripted based on the meeting number it is for that agent. For example, if a meeting happened between agents A and B, where this was A's first meeting and B's third the node name would be $\{A_1, B_3\}$. Naming in this manner allows for meeting graph synchronization when two agents meet.

When two agents meet their meeting graphs are synchronized to update the mutual beliefs held by the two agents. Meeting graphs are combined by synchronizing the two agent's current meeting graphs with each other and then adding a new node representing the current meeting that connects to the most recent meeting of each of the two agents. In this way a node will only be preceded by two other nodes, representing the last meetings of the two currently meeting agents. During the synchronization step both agent's meeting graphs are projected onto each other so that there is no duplicate information. To represent this consider the case where agent A and agent B have an initial meeting, and then independent of each other both agents meet with a unique agent, i.e. agent A meets with agent C and agent B meets with agent D. In this case both

agent A and agent B have meeting graphs that contain the same initial node, but then this node connects to a different node in both graphs which represent the second meeting the agent participated in. The synchronization of this would be to only keep one copy of the initial meeting between agent A and B, but connect it to both subsequent meetings.

Using this meeting graph to reason about an agent's beliefs is simple once the synchronization step is complete. To determine an agent's beliefs an agent must simply traverse the graph from the most recent meeting breadth-first until a node is found containing the agent that is to be reasoned about. The shallowest node in the tree containing an agent is the most current set of the beliefs that meeting graph contains about that particular agent, since a node represents the belief of that agent at that meeting. In order for an agent to determine its own beliefs it can simply look at the most recent node, which contains all the facts that are currently believed by that agent.

Goal Detection

At the end of each agent's perception stage during a simulation iteration a goal test is performed. To determine if the goal has been satisfied an agent consults its meeting graph to determine if all objects have been collected. This is done by looking at the last node added to the meeting graph and determining if the set of beliefs contains the collection of all objects. If it is determined all objects have been found then the agent will begin the process of returning to the start. In order for the simulation to end all objects must be collected and all agents must return to the starting point. In order not to leave any agents behind when an agent determines that the goal is complete this agent then continues the search strategy until it encounters another agent. Once it encounters another agent it will pass along the message that it is returning to the start. On subsequent

movement steps the agent then begins returning directly to the starting point. The agent that the message was passed to then begins the process the initial agent started and seeks out another agent to inform that the goal is complete. Once another agent is found the message is passed along and the agent returns to the start. This pattern is repeated until all agents return to the start and the goal is complete. This method becomes increasingly inefficient the fewer agents remain in the world, however it guarantees that all agents will eventually return and avoid the General's Problem where one agent could be stranded in the world with no way to gain the belief that all objects are collected.

CHAPTER IV

RESULTS AND DISCUSSION

Experiment Strategy

Several tests have been run on the simulation environment to discover the efficiency of the proposed search strategies compared to the random walk search strategy. Each experiment featured a sample of multiple runs of each strategy with a fixed number of agents and objects, but with variable sensor range, and environment dimensions. The focus of these experiments was to determine which search strategy is the most efficient – completes the task in the shortest time – and if varying the sensor range, or search space size affects that efficiency relative to the other search strategies.

In each experiment fifty trials were performed for each search strategy and then these trial sets were compared to one. During each trial the time taken to fully complete the goal will be recorded.

The experiments were broken up into three sets: small environment, medium environment, and large environment. The three environment sizes are aimed at determining if a search strategy is more efficient in one environment size and less efficient in another relative to the other strategies. Agent sensor range was also changed in each experiment set to determine if a search strategy performs better or worse with different agent communication ranges. Sensor ranges were set at one, two, and three units away from the agent to simulate more and less limited communication.

Experiments

Small environment

In this experiment set, the size of the world is limited to fifty units square. Table I shows a summary of the simulation results obtained from running all search strategies in a small environment with varying sensor ranges; Figure I shows this summarized information in chart form.

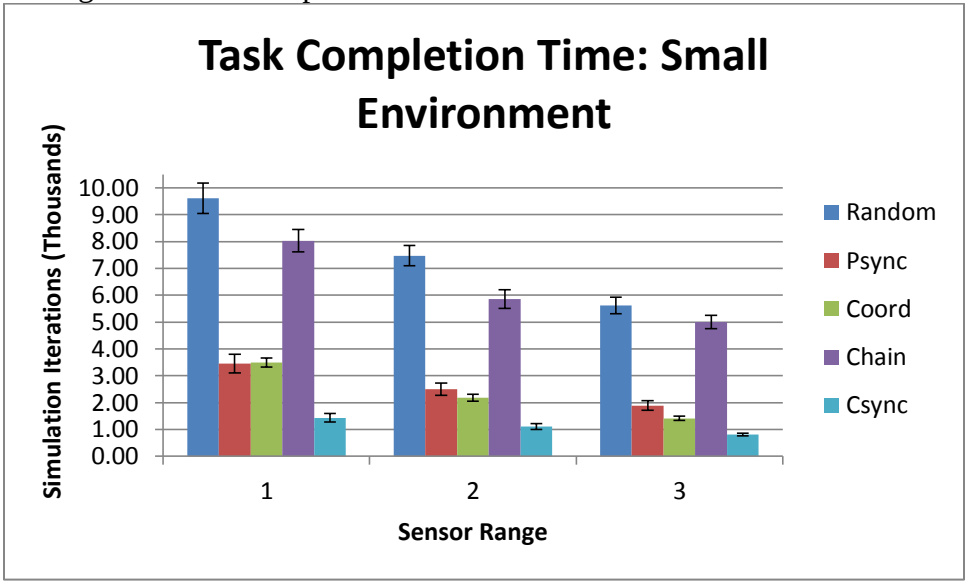
Table I. Summary of Small Environment Simulations

Strategy	Sensor Range	Average Time Steps	Standard Deviation
Random	1	9615.72	4027.04
Random	2	7473.88	2683.69
Random	3	5614.96	2228.36
Psync	1	3450.24	2513.73
Psync	2	2493.38	1625.87
Psync	3	1890.26	1296.91
Coord	1	3487.20	1220.91
Coord	2	2177.28	943.15
Coord	3	1409.34	587.98
Chain	1	8029.08	2934.62
Chain	2	5858.60	2479.53
Chain	3	5002.92	1782.98
Csync	1	1434.86	1154.98
Csync	2	1111.46	792.33
Csync	3	815.48	372.97

This table summarizes the results of simulations run on the small environment size of fifty square units. The search strategies used were: Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

As expected the Random search strategy performed worse than the Periodic Synchronization, Circulating Coordinator, and Coordinator-Synchronization search strategies. Unexpectedly the Chain Model performed significantly worse than all other strategies besides the Random Walk. These results can be seen more clearly in Figure I.

Figure I: Task Completion Time of Small Environment Simulations



This chart shows the results of simulations run on the small environment size of fifty square units. The search strategies used were: Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

Medium environment

In this experiment set, the size of the world is limited to one hundred units square. Table II shows a summary of the simulation results obtained from running all search strategies in a medium environment with varying sensor ranges as well as the average increase in time steps from the same search strategy and sensor range combination from the medium environment experiments; Figure II shows this summarized information in chart form.

Table II. Summary of Medium Environment Simulations

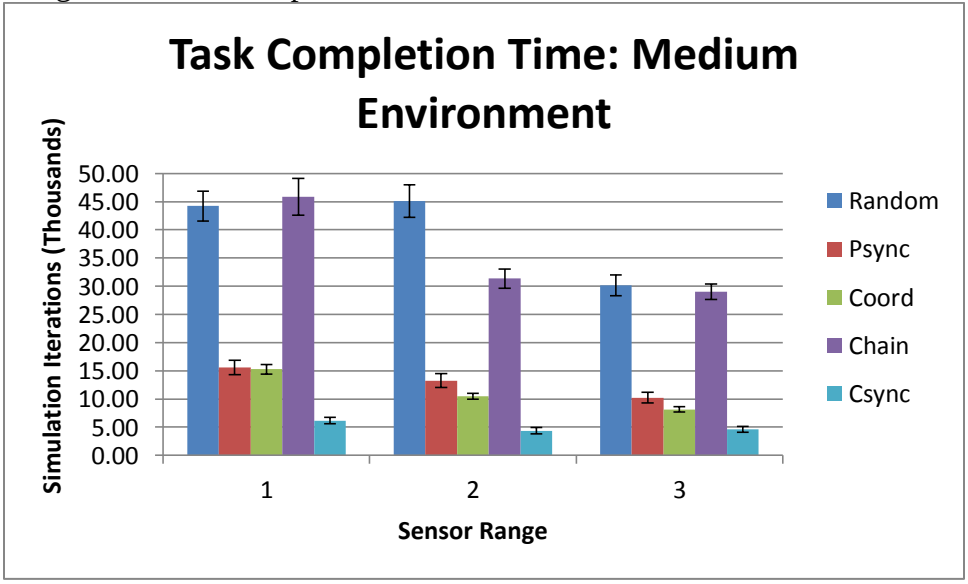
Strategy	Sensor Range	Average Time Steps	Standard Deviation	Avg Time Step Increase
Random	1	44221.48	18602.48	359.89%
Random	2	45130.72	20426.07	503.85%
Random	3	30125.72	13054.00	436.53%
Psync	1	15577.60	8935.01	351.49%
Psync	2	13257.68	8587.15	431.72%
Psync	3	10231.94	6686.71	441.30%
Coord	1	15268.86	5844.30	337.85%
Coord	2	10474.64	3852.36	381.09%

Coord	3	8152.70	3067.52	478.48%
Chain	1	45864.96	23181.32	471.24%
Chain	2	31361.72	12126.77	435.31%
Chain	3	29004.44	9624.82	479.75%
Csync	1	6167.10	3929.79	329.80%
Csync	2	4346.22	3881.99	291.04%
Csync	3	4620.40	3722.71	466.59%

This table summarizes the results of simulations run on the medium environment size of one hundred square units. The search strategies used were: Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

These results follow a similar trend to the results obtained from the small environment simulations, except the differences between strategies is more pronounced, as seen in Figure II. It was unexpected, however, that the Chain Model for the lowest sensor range test would perform as close to the Random Walk as it did, performing worse on average.

Figure II: Task Completion Time of Medium Environment Simulations



This chart shows the results of simulations run on the medium environment size of one hundred square units. The search strategies used were: Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

Large environment

In this experiment set, the size of the world is limited to two hundred units square. Table III shows a summary of the simulation results obtained from running all search strategies in a large

environment with varying sensor ranges as well as the average increase in time steps from the same search strategy and sensor range combination from the large environment experiments; Figure III shows this summarized information in chart form.

Table III. Summary of Large Environment Simulations

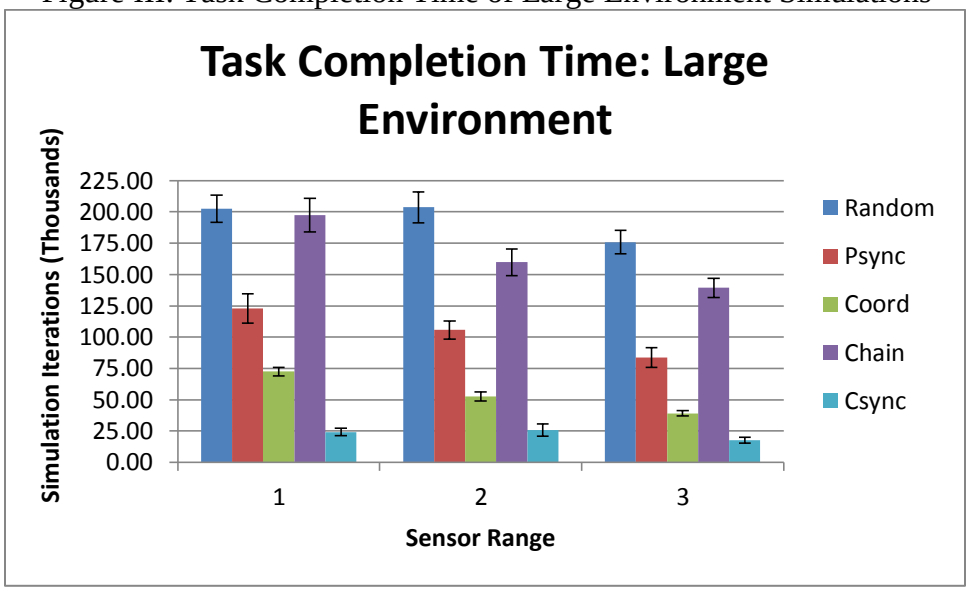
Strategy	Sensor Range	Average Time Steps	Standard Deviation	Avg Time Step Increase
Random	1	202490.96	76441.38	357.90%
Random	2	203636.20	88067.39	351.21%
Random	3	175886.12	66666.53	483.84%
Psync	1	123057.02	83340.01	689.96%
Psync	2	105823.26	50865.16	698.20%
Psync	3	83692.98	55631.86	717.96%
Coord	1	72562.36	24215.79	375.23%
Coord	2	52642.78	24510.65	402.57%
Coord	3	39212.98	15582.52	380.98%
Chain	1	197314.00	94389.58	330.21%
Chain	2	159781.68	75642.16	409.48%
Chain	3	139326.12	55104.99	380.36%
Csync	1	24300.94	21545.97	294.04%
Csync	2	25851.12	33466.43	494.80%
Csync	3	17928.88	16262.38	288.04%

This table summarizes the results of simulations run on the large environment size of two hundred square units. The search strategies used were: Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

Similar to the medium environment size simulations, these results show a similar trend to the small environment simulations, however instead of the Periodic Synchronization strategy remaining fairly close in completion time to the Circulating Coordinator strategy it proved to do significantly worse.

From these results conclusions can be drawn that will be able to determine which search strategy is the most efficient and if the initial expectations about the search strategies held true across the simulations.

Figure III: Task Completion Time of Large Environment Simulations



This chart shows the results of simulations run on the large environment size of two hundred square units. The search strategies used were: Random Walk (Random), Periodic Synchronization (Psync), Circulating Coordinator (Coord), Chain Model (Chain), and Coordinator-Synchronization (Csync).

Discussion

These results show that as a search strategy becomes more centralized it trends towards being more efficient. The one outlier in this analysis, however, is the Chain Model search strategy, that is centralized in that each agent is assigned a specific area to search, but moves randomly outside of that restraint. This poor performance is the result of the necessity for all agents to believe that the task is complete before returning to the start. Because of this, and because there is no centralized way for the agents to communicate it leaves the information dissemination to happen much more similarly to the Random Walk strategy, which greatly hinders overall performance, even if all the objects were collected in an amount of time more similar to the other models that divide the search space.

From the collected data it is also clear that the prediction of the Random Walk performing relatively worse as the search space increase was false, and it in fact performed similarly in all

three environment sizes. It is also clear that while there was only slight variation in relative completion time for the Periodic Synchronization strategy when moving from the small to medium environment, moving from the medium to large environment greatly impacted the strategy's performance. From the data collected the conclusion can be drawn that the strategy most affected by environment size increases in the Periodic Synchronization strategy, which is most likely caused by the agents being forced to meet in the middle of the environment before resuming their search. This would cause the fringes of the search space to be search less frequently and could stop the agents from collecting the objects located there.

The search strategy that performed the best overall, completing the task in the shortest average time steps, was the Coordinator-Synchronization strategy. Intuitively this makes sense as it combines the best aspects of the Circulating Coordinator and Periodic Synchronization strategies, which allows for a more even search pattern across the entire environment to speed up object collection, and the periodic meetings to spread this information among all the agents quicker than just the coordinator can accomplish.

Many simplifications were made to the problem of distributed agent search to test these search strategies, and these issues will need to be addressed before any of these strategies are applied in a real world environment.

A real world environment that must be search, especially one with limited communication, is likely to not be as simple as a two-dimensional plane. Because of this the strategy that is used would need to be modified to fit the environment, such as dividing the search space into sections

and sending a group of agents into each section, with some way to distribute beliefs between sections.

Another issue to address is that once an agent collects an object, in the simulation it continues to wander around searching for other objects, even though it cannot carry another one. This could be changed so that the agent is performing some other useful task when it is in this state.

The last major issue is the simplicity of the decision making of the agents. In a real world environment the agents will have many other decisions to make besides the movement direction to make and this will need to be added in on top of the search strategy.

From here the next step of research would be to take the search strategies, specifically the Coordinator-Synchronization strategy and add additional logic in order to attempt to increase its performance. Additions could include a more directed search pattern where agents keep track of uncollected objects they detect and are able to spread that information so other agents have more information when choosing where to search, or a method that allows agents to swap assigned positions or assignments, such as a searching agent becoming the new coordinator, in order to give agents who still need to collect objects, such as the coordinator, a chance to do this is a more directed manner.

CHAPTER V

CONCLUSIONS

As autonomous agents are tasked with completing ever more difficult search tasks, efficient search strategies that they can implement will become increasingly necessary. Because of this and the possibility that these new environment will require a shared belief state between the agents that must persist even in the event of non-constant communication new search strategies need to be developed.

In this thesis I have proposed and tested the efficiency of several search strategies that are designed for this non-constant communication environment where agents must be able to reason about the beliefs of other agents. Of the proposed strategies there was one that outperformed the others, which was the Coordinator-Synchronization strategy. This strategy was shown to be both the least affected by environment size increase, and the least affected by communication range.

RERFERENCES

[Doyle, 1978] Doyle, J. (1978) Truth Maintenance Systems for Problem Solving. AITR-419.

[Tambe, 1997] Tambe, M. (1997) Agent architectures for flexible, practical teamwork. National Conference on Artificial Intelligence (AAAI-97), pages 22-28.

[Levesque *et al.*, 1990] Levesque, H. *et al.* (1990) On Acting Together. National Conference on Artificial Intelligence (AAAI-90), pages 94-99.

[Shoham and Leyton-Brown, 2009] Shoham, Y.; Leyton-Brown, K. (2009) Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. New York: Cambridge University Press, pages 409-436.